Differential forms for target tracking and aggregate queries in distributed networks

Resilient Routing for Sensor Networks Using Hyperbolic Embedding of Universal Covering Space

Wei Zeng* Rik Sarkar* Feng Luo† Xianfeng Gu* Jie Gao*

* Department of Computer Science, State University of New York at Stony Brook. {zengwei, rik, gu, jgao}@cs.sunysb.edu
† Mathematics Department, Rutgers University. fluo@math.rutgers.edu

***Abstract*—We study how to characterize the families of paths between any two nodes s, t in a sensor network with holes. Two paths that can be deformed to one another through local changes are called *homotopy equivalent*. Two paths that pass around holes in different ways have different *homotopy types*. With a distributed algorithm we compute an embedding of the network in hyperbolic space by using Ricci flow such that paths of different homotopy types are mapped naturally to paths connecting s with different images of t. Greedy routing to a particular image is guaranteed with success to find a path with a given homotopy type. This leads to simple greedy routing algorithms that are resilient to both local link dynamics and large scale jamming attacks and improve load balancing over previous greedy routing algorithms.**

I. Introduction

This paper is motivated by routing algorithm designs that are resilient to dynamics in a sensor network. In a typical large scale sensor network, there are network changes of different scales. At the node level resolution, wireless links could undergo sporadic changes. Link quality varies over time. Nodes may fail. Packets may get lost due to wireless interference as in the hidden terminal problem. At a larger scale, communication links in a region can be temporarily disabled by jamming attacks, either imposed by malicious parties [25], or as a result of co-located multiple benign wireless networks interfering with each other. For example, experiments have shown that 802.15.4 sensor network interferes with existing WiFi network resulting in 54% packet loss [15]. In both cases, it is unpredictable whether a packet is able to go through along a predetermined path. We need resilient routing schemes that can tolerate such sudden changes of link quality and have flexibility to reactively choose one from many possible paths to the destination.

For a source s and a destination t there are many different paths from s to t. Explicitly storing all of them, for all possible source destination pairs, is clearly not feasible in a resource constrained sensor network. Previous work on routing resilience has mainly focused on heuristic algorithms for multi-path routing [7]. Randomization may also be used in the routing metric to introduce some path diversity. However, it is still unclear how to evaluate the resilience of a set of paths obtained this way. How do we know that we take a path 'sufficiently far away' from the previous one? In this paper we would like to study in depth the characteristics of the 'space of paths': the classification of paths and design of light-weight routing schemes that can easily navigate in this space of paths, leading to improved resilience to failures at different scales.

In this paper we focus on large networks with non-uniform sensor distribution. The network can have a complex shape with multiple holes. As networks grow large in size and terrain variation and obstacles/landscape features forbid sensor placement, it is unrealistic to assume that the sensors are always uniformly deployed in a region of some regular shape. Furthermore, the problem of resilient routing in a multi-hole network is particularly challenging – intuitively this involves some relatively global decisions such as whether we should route from the left of the hole or from the right of the hole.

Path homotopy types. Let us look at the example in Fig. 1. There are three holes in the network and there are many different ways to route from s to t. Observe that the paths α, β, γ are all different in a global sense. They get around the three holes in different ways. One can not deform α to β unless it jumps over some hole. However, paths γ and δ are only different in a local manner. One can deform γ to δ smoothly through some local changes. This difference is characterized by the *homotopy type* of a path. Two paths in a Euclidean domain are *homotopy equivalent* if one can smoothly deform one to the other. Paths that are pairwise homotopy equivalent are said to have the same homotopy type. The number of homotopy types is infinite, as one can tour around a hole k times, with any integer k. But for most routing scenarios we only care about a small number of homotopy types.

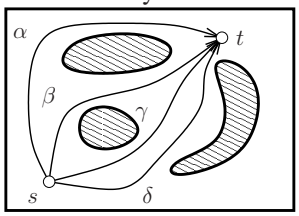
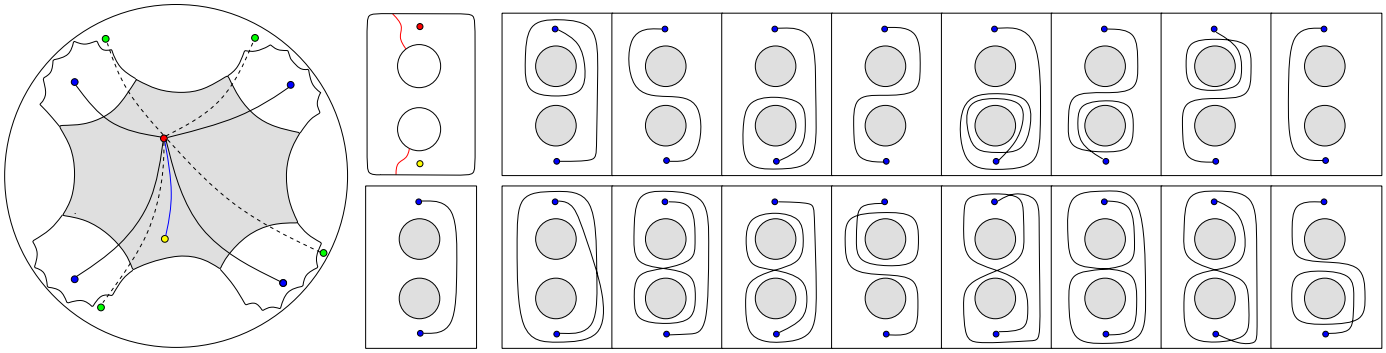


Fig. 1. The network has three holes shaded. Paths α, β, γ all have different homotopy type. γ and δ are homotopy equivalent.

Understanding the homotopy types of the paths from s to t is important for load balancing and resilience. For example, sporadic link dynamics (unless they create a hole) can be possibly avoided by taking a slightly different path with the same homotopy type. But to get around a large jamming attack that destroys a 'bridge' of the network we will have to take a path of different homotopy type.

Now the question is, how to compute the homotopy type of a given path? How to tell two paths are homotopy equivalent or not? How to choose a path that has a different homotopy type from the previous one? How to find the shortest path of a given homotopy type? For all these questions we need a compact way to encode the homotopy type of all the paths, and a distributed, local algorithm to find a path of a given homotopy type.

Our approach. Our solution is to embed the given network



(a) portion of UCS (b) π_1-group (c) shortest paths induced by 2-ringed UCS

Fig. 2. Computing the shortest path on the hyperbolic embedding of universal covering space for a 3-connected domain S. S is embedded to hyperbolic polygon on the Poincaré disk in (a) with the homotopy group (π_1-group) shown in the top frame of (b) with homotopy group generators $\{a_1, a_2\}$. Given the source s and target t on S, the family of the shortest path γ from s to t is computed as the geodesics $\tilde{\gamma}$ from s to the images of t on universal covering space (UCS). In theory, there are infinite shortest paths from s to t. The bottom frame of (b) shows the shortest path within one fundamental domain, which differs from frame (8) in (c) by a word $a_1 a_1^{-1} a_2 a_2^{-1}$. (c) continues to illustrate the other shortest paths under the 2-ringed UCS setting. Notice that frames (1-4) can also be given by the 1-ringed UCS, which has provided rich information for greedy routing. [26]

in *hyperbolic space*. We first compute a triangulation of the network from the connectivity graph by a distributed and local algorithm developed in [22]. Holes are modeled as non-triangular faces. The holes in the network are cut open to get a simply connected triangulation. Let us call it T. Using the Ricci flow algorithm (to be explained later) we embed T in a convex region S in hyperbolic space. Each node is given a hyperbolic coordinate. Each edge uv has a length $d(u, v)$ as the geodesic between u, v in the hyperbolic space. In this way, greedy routing with the hyperbolic metric, i.e., send the message to the neighbor closer to the destination measured by hyperbolic distance, has *guaranteed delivery*.

In fact, with the Ricci flow algorithm we can get the embedding of the triangulation T to infinitely many patches. Each patch is a convex piece and the patches are congruent (isometric) to each-other. Two patches can be transformed to one another by a suitable isometry-preserving transformation in the hyperbolic plane. The patches are glued naturally along their shared boundaries and they tile an unbounded portion of the hyperbolic plane. This is called the *universal covering space* of the topology of the network. We fix the source $\hat{s}_1$ in one patch $\hat{T}_1$, and take the image of the destination $\hat{t}_i$ in patch $\hat{T}_i$. All the paths that connect s to t with the same homotopy type will map to the paths that all connect $\hat{s}_1$ to the same image $\hat{t}_i$. Thus, if we would like to get a path of different homotopy type from the previous one, we can simply use greedy routing to find one between $\hat{s}_1$ and a different image $\hat{t}_j$. In other words, the paths of different homotopy type are compactly coded by different images of the destination in the embedding. Although there are theoretically infinitely many images (and infinitely many homotopy types for paths connecting s and t), all these cases can be generated from a small amount of information. For practical settings we only care about a constant number of different homotopy types (paths that surround a hole many times are not interesting). Thus we need to be concerned about only the corresponding patches in the covering space.

Recall that the hyperbolic embedding is convex. That is, all the hole boundaries are on the outer boundary of the embedding, and the shortest path of a particular homotopy type between any two points will *not* pass through a boundary unless either the source or the destination is on the boundary. This is different from many greedy routing schemes that route 'around holes' by following the hole boundaries [1], [3], [13], [22], with which boundary nodes necessarily carry more traffic. Thus our scheme has better load balancing than previous schemes, as demonstrated by simulation results.

The universal covering space can be used to find loops of different homotopy types when $s = t$. For any point s, we can find its images in different patches $\hat{s}_1$, $\hat{s}_i$, $i \neq 1$. The path connecting $\hat{s}_1$, $\hat{s}_i$ is a loop in the original triangulation. The paths connecting $\hat{s}_1$ with different other images $\hat{s}_i$ have different homotopy types (i.e., surrounding different set of holes). This can be useful for the applications when we want to find a loop surrounding a target hole or multiple target holes. Or, given any target hole, test whether a group of sensors successfully surround it (mathematically speaking, cycle contractibility). If we want to count the number of people entering/leaving a building, we only need to activate a loop of sensors surrounding the building and aggregate their detections. As there can be many different loops by choosing different s and different paths connecting two images of s, we can have different loops taking turns to accomplish the sensing task.

To summarize, we compute an embedding of a given network in the hyperbolic space such that

1) Delivery is guaranteed by greedy routing.
2) Paths of different homotopy types are grouped as paths connecting the source to different images of the destination in the embedding. One can easily use greedy routing to select paths of different homotopy types.
3) The greedy path (i.e., the shortest path in the hyperbolic metric) does not go through any boundary node unless the source or destination is on the boundary. This means that the boundary nodes are not getting more load, as is typical in many greedy routing schemes.

The embedding is computed through a distributed, iterative algorithm using Ricci flow, which was introduced by Richard

Hamilton for Riemannian manifolds of any dimension in his seminal work [9] in 1982. Intuitively, on a Riemannian surface the *Riemannian metric* specifies the length of any curve on the surface, which then determines the *Guassian curvature* at any point. A surface Ricci flow is a process to deform the Riemannian metric of the surface, in proportion to Gaussian curvatures, such that the curvature evolves in the same manner as heat diffusion. It is a powerful tool for finding a Riemannian metric satisfying the prescribed Gaussian curvature and has been applied in the proof of the Poincare conjecture on 3-manifolds [16]–[18]. Chow and Luo [2] extended the idea to a discrete triangulated surface and proved a general existence and convergence theorem for the discrete Ricci flow. Jin *et.al.* provided an algorithm in [11]. In our case, we use Ricci flow to deform the network such that all the interior vertices have curvature -1 (thus being flat on a hyperbolic plane) and vertices on boundary have curvature 0 (thus following a hyperbolic straight line). The network is mapped to a convex piece in the hyperbolic space and is exactly what we need for routing.

Related work on greedy routing. Greedy routing has been extensively studied in sensor networks. The greedy criterion can be minimizing distance to the destination, measured by geographical coordinates [1], [13] or virtual coordinates of an embedding of the network in some space [19], [22]. Most of these greedy methods do not guarantee delivery [1], [13], [19]. In our previous work, we used Euclidean Ricci flow to embed the network in the Euclidean plane such that all the network holes are circular and greedy routing has guaranteed success. In another work, Flury *et.al.* shows a greedy scheme to find paths of bounded stretch [4], with an embedding in $O(\log n)$ dimensional Euclidean space. Embedding of the network in hyperbolic space has also been proposed by Kleinberg [14]. He shows that any tree can be embedded in a hyperbolic space such that greedy routing works on the tree. This is used to show that any graph has an embedding in hyperbolic space that admits greedy routing.

None of the greedy methods above is able to find paths of different homotopy types, which is the focus of this paper.

II. Theoretic Background

This section briefly introduces the theoretic background necessary for the current work. For details, we refer readers to [21] for algebraic topology and [23] for differential geometry.

A. Homotopy Group and Universal Covering Space

Let S be a topological surface, and p be a point of S. All loops with base point p are classified by homotopy relations. All homotopy equivalence classes form the *homotopy group* or *fundamental group* $\pi_1(S,p)$, where the product is defined as the concatenation of two loops through their common base point.

Suppose S is of genus zero with $n+1$ boundaries, $\{b_0, b_1, \cdots, b_n\}$, where b_0 is the exterior boundary, $b_k, k > 0$ are interior ones, then S is called a $n+1$ connected domain, or simply a *multiply connected domain*. Figure 1 shows a 4-connected domain. In the following discussion, we always assume $n > 1$. $\pi(S,p)$ is a free group generated by $\{a_1, a_2, \cdots, a_n\}$, where a_k goes around the kth boundary b_k.

A *covering space* of S is a space $\tilde{S}$ together with a continuous surjective map $h : \tilde{S} \to S$, such that for every $p \in S$ there exists an open neighborhood U of p such that $h^{-1}(U)$ (the inverse image of U under h) is a disjoint union of open sets in $\tilde{S}$ each of which is mapped homeomorphically onto U by h. The map h is called the *covering map*. A simply connected covering space is a *universal covering space (UCS)*.

A *deck transformation* of a cover $h : \tilde{S} \to S$ is a homeomorphism $f : \tilde{S} \to \tilde{S}$ such that $h \circ f = h$. All deck transformations form a group, the so-called *deck transformation group*. A *fundamental domain* of S is a simply connected domain, which intersects each orbit of the deck transformation group only once.

Suppose $\gamma \subset S$ is a loop through the base point p on S. Let $\tilde{p}_0 \in \tilde{S}$ be a preimage of the base point p, $\tilde{p}_0 \in h^{-1}(p)$, then there exists a unique path $\tilde{\gamma} \subset \tilde{S}$ lying over γ (i.e. $h(\tilde{\gamma}) = \gamma$) and $\tilde{\gamma}(0) = \tilde{p}_0$. $\tilde{\gamma}$ is a *lift* of γ.

The deck transformation group $Deck(S)$ is isomorphic to the homotopy group $\pi_1(S,p)$. Let $\tilde{p}_0 \in h^{-1}(p)$, $\phi \in Deck(S)$, $\tilde{\gamma}$ is a path in the universal cover connecting $\tilde{p}_0$ and $\phi(\tilde{p}_0)$, then the projection of $\tilde{\gamma}$ is a loop on S, ϕ corresponds to the homotopy class of the loop, $\phi \to [h(\tilde{\gamma})]$. This gives the isomorphism between $Deck(S)$ and $\pi_1(S,p)$.

The whole UCS is tessellated by fundamental domains, denoted as D_k's. One fundamental domain D_0 is selected as the central fundamental domain. Any fundamental domain D_k differs from D_0 by a deck transformation ϕ_k, which corresponds to a unique homotopy class $[\gamma_k] \in \pi_1(S,p)$. The *n-ringed UCS* is the union of all D_k's , such that the length of the shortest word of $[\gamma_k] \in \pi_1(S,p)$ is no greater than n. Figure 2 shows the 2-ringed UCS on $\mathbb{H}^2$ for a 3-connected domain.

B. Uniformization Metric

Let S be a surface embedded in $\mathbb{R}^3$. S has a Riemannian metric induced from the Euclidean metric of $\mathbb{R}^3$, denoted by $\mathbf{g}$. The total Gaussian curvature of S is solely determined by the topology of the surface, as shown below.

Theorem 2.1 (Gauss-Bonnet): The total Gaussian curvature is given by $\int_S K dA + \int_{\partial S} k_g ds = 2\pi\chi(S)$, where K is the Gaussian curvature on interior points, k_g is the geodesic curvature on boundary points ∂S, $\chi(S)$ is the Euler characteristic number of S.

Suppose $u : S \to \mathbb{R}$ is a scalar function defined on S. Then $\bar{\mathbf{g}} = e^{2u}\mathbf{g}$ is also a Riemannian metric on S and is conformal to the original one. Any surface admits a Riemannian metric of constant Gaussian curvature, which is conformal to the original metric. Such metric is called the *uniformization metric*.

Theorem 2.2 (Uniformization): Suppose S is a $n+1$ connected domain with a Riemannian metric $\mathbf{g}$, $n > 1$, then there exists a Riemannian metric $\tilde{\mathbf{g}}$, such that $\tilde{\mathbf{g}}$ induces -1 Gaussian curvature at every interior point of S, 0 geodesic curvature at every boundary point. Furthermore, $\tilde{\mathbf{g}}$ is conformal to $\mathbf{g}$.

C. Poincaré Disk Model and Hyperbolic Uniformization

In this work, we use Poincaré disk to model the hyperbolic space $\mathbb{H}^2$, which is the unit disk $|z| < 1$ on the complex plane with the metric

$$ds^2 = \frac{4dzd\bar{z}}{(1-z\bar{z})^2}.$$

In this model the isometry-preserving transformations of the plane are given by Möbius transformations of the form:

$$z \to e^{i\theta}\frac{z-z_0}{1-\bar{z}_0 z},$$

where θ and z_0 are parameters. The geodesics, or hyperbolic lines, are circular arcs perpendicular to the unit circle. Let z_1 and z_2 be two points inside the Poincaré disk, then there exits a unique geodesic passing through z_1 and z_2. See Fig. 3 (a). Suppose the geodesic intersects the unit circle at ξ_1, ξ_2, ξ_1 is closer to z_1 and ξ_2 is closer to z_2, then hyperbolic distance between z_1, z_2 is given by $d(z_1, z_2) = \log[z_1, z_2; \xi_1, \xi_2]^{-1}$, where the complex cross ratio $[z_1, z_2; \xi_1, \xi_2] = \frac{(z_1-\xi_1)(z_2-\xi_2)}{(z_2-\xi_1)(z_1-\xi_2)}$ is a real number, because four points z_1, z_2, ξ_1, ξ_2 are on the same circle.

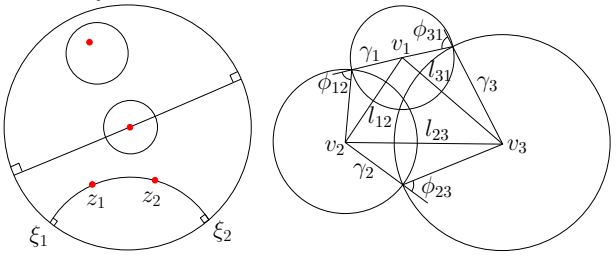
Suppose S is a multiply connected genus zero surface with the hyperbolic uniformization metric $\tilde{\mathbf{g}}$. Then its universal covering space $(\tilde{S}, \tilde{\mathbf{g}})$ can be isometrically embedded in $\mathbb{H}^2$. Any deck transformation of $\tilde{S}$ is a Möbius transformation, and called a *Fuchsian transformation*. These transformations form a group called the *Fuchsian group* of S.

The following properties about the hyperbolic uniformization metric of a multiply connected domain can be deduced from Theorem 2.2. These are useful for routing in sensor networks.

Corollary 2.3 (Convexity): Suppose S is a multiply connected domain with the hyperbolic uniformization metric, p and q are two points on S. In each homotopy class, the geodesic connecting p and q exists, and is unique.

Similarly, the universal covering space $\tilde{S}$ with the hyperbolic metric $\tilde{\mathbf{g}}$ is also convex, its boundaries become hyperbolic lines (geodesics). In hyperbolic space $\mathbb{H}^2$, two points determine a unique hyperbolic line (geodesic). Two lines intersect at a single point. Therefore,

Corollary 2.4: If a geodesic connecting p and q in $(\tilde{S}, \tilde{\mathbf{g}})$ intersects the boundary $\partial\tilde{S}$, then at least one of p or q is on the boundary.



(a) hyperbolic geometry (b) circle packing metric

Fig. 3. Hyperbolic geometry and hyperbolic embedding.

Given two points p and q on a multiply connected domain S, fix a path γ_0 from p to q. Let γ be another path from p and q, then the concatenation $\gamma\gamma_0^{-1}$ is a loop with the base point q, we say the homotopy type of $\gamma\gamma_0^{-1}$ in $\pi_1(S, q)$ is the homotopy type of γ. By this way all paths from p to q are classified by homotopy.

Corollary 2.5 (Shortest Path): Let S be a multiply connected domain with hyperbolic metric. Given two points p and q on S, then for each homotopy class in $\pi_1(S, p)$, there exists a unique geodesic from p to q.

Namely, given a word in $\pi_1(S)$ representing a homotopy type, one can find the unique geodesic of that type from p to q.

Figures 2 shows the case of 3-connected domain for computing the shortest paths from the geodesics on hyperbolic UCS. Each geodesic on UCS is projected to a shortest path on the original domain. Each of them has different homotopy type, which is determined by the homotopy word. Figure 6 shows two shortest paths with different homotopy types in 1-ring of universal covering space.

D. Ricci Flow

Ricci flow is a powerful curvature flow method, invented by Hamilton for the proof of the Poincaré conjecture [10]. Intuitively, Ricci flow is the process to deform the Riemannian metric according to the curvature, such that the curvature evolves like a heat diffusion process:

$$\frac{dg_{ij}}{dt} = -2K + \frac{\chi(S)}{A},$$

where K is the Gaussian curvature induced by the metric $\mathbf{g}(t)$, A is the area of the surface. For closed surfaces with non-positive Euler numbers, Hamilton proved the convergence of Ricci flow in [10]:

Theorem 2.6 (Hamilton 1988): For a closed surface of non-positive Euler characteristic, if the total area of the surface is preserved during the flow, the Ricci flow will converge to a metric such that the Gaussian curvature is constant everywhere.

In our case, a multiply connected domain is not closed. However, we can glue two copies of the same multiply connected domain along their common boundaries to form a symmetric high genus surface. By applying Hamilton's Ricci flow we can get the hyperbolic metric everywhere. Furthermore, because of the symmetry, the geodesic curvatures along the boundaries of the original surface will become zero.

Corollary 2.7: For a $n+1$ connected domain, if the total area of the surface is preserved during the flow, the Ricci flow will converge to a metric such that the Gaussian curvature is constant on interior points and the geodesic curvature is zero on the boundary points.

Thus, we can compute the hyperbolic uniformization metric using surface Ricci flow method, as in this paper.

III. Hyperbolic Embedding Algorithms

This section shows how to obtain a hyperbolic embedding of the universal covering space. In the next section we will show how to use the hyperbolic embedding for greedy routing realizing paths of different homotopy types.

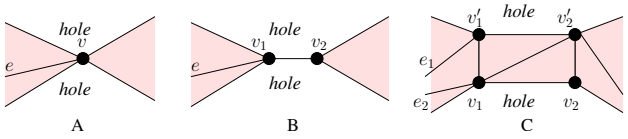


Fig. 4. Handling degeneracy. (A) Holes sharing a boundary vertex, (B) Holes sharing a boundary edge and (C) triangulation using virtual nodes.

A. Triangulation of Sensor Networks

To obtain the hyperbolic embedding, we first need a triangulation from the connectivity graph of a sensor network. In our previous work [22] we have proposed distributed algorithms to extract a triangulation from a unit disk graph (UDG) or a quasi unit disk graph. All non-triangular faces are interior holes.

Location-based triangulation. When the nodes have their locations, we can compute a triangulation as the restricted Delaunay graph (RDG) [8]. In particular, each node computes the Delaunay triangulation of its 1-hop neighborhood. Neighboring nodes exchange their local Delaunay triangulations and remove edges that are invalidated in any local Delaunay triangulation. In [22] we have extended the method to quasi-unit disk graphs with parameter $1 \leq \alpha \leq \sqrt{2}$. This is a graph where nodes less than $1/\alpha$ away always have an edge, while nodes between $1/\alpha$ and 1 away may not have an edge. The idea is to compute the RDG with $1/\alpha$ sized disks for neighborhoods instead of unit disk neighborhoods, which produces a planar graph that may not be connected. Then virtual edges are included to restore the connectivity. More details can be found in [22].

Landmark-based triangulation. When the nodes do not have locations, we can use a landmark-based scheme [5], [6] to come up with a triangulation of the sensor field. First, a subset of nodes are selected as landmarks in a distributed manner. The landmarks are uniformly distributed such that any two landmarks are k hops apart (for a small $k = 5$ or 6), and any non-landmark node is within k hops of some landmark. This method requires a flood from a landmark to last only k hops, hence the overall cost is linear for a network of bounded density. Each node then selects its closest landmark and nodes closest to the same landmark are grouped in to a Voronoi cell. The adjacency of these cells give rise to a dual *combinatorial Delaunay complex (CDC)*. Crossing edges in the CDC can be properly handled as shown in [5], [6] so that the resulting graph is planar.

Handling degeneracies. The hyperbolic embedding algorithm requires an input as a triangulated 2D manifold. The triangulation we obtained above may have all kinds of degeneracies. For example, two holes are adjacent at a vertex, or two holes share a common boundary as a chain of nodes. See Fig. 4 (A-B). We can add virtual nodes to eliminate such degeneracies as shown in Fig. 4 (C).

Slicing the holes open. Once we obtain a triangulation, all non-triangular faces are holes, and their boundaries together with the outer boundary form the boundary components. We choose an arbitrary boundary component β (say, the longest). Then from each other component we flood to find a simple path to β. By a simple exchange process, it is easy to ensure that any pair of such paths do not cross. Next we cut open each hole by cutting along the corresponding path. In this process, each node on the path is split into two virtual nodes - one for each side of the path. When this process is completed, the domain is *simply connected* and contains no hole. In fact, it is not necessary for the cut paths to always connect to the same boundary, as long as the final result is simply connected. For example, a cut locus can determine the cuts [24]. However we will stay with the current method for simplicity of presentation.

B. Discrete Hyperbolic Ricci Flow

Given the triangulation M extracted from the sensor network connectivity graph, with $\{v_1, v_2, \cdots, v_n\}$ as the vertex set, $[v_i, v_j]$ be an edge connecting v_i and v_j, $[v_i, v_j, v_k]$ be a triangular face, the *discrete metric* of M is the edge length metric. Let θ_i^{jk} be the corner angle at vertex v_i in the face $[v_i, v_j, v_k]$. We treat each face $[v_i, v_j, v_k]$ as a hyperbolic triangle, therefore θ_i^{jk} is determined by the edge lengths using hyperbolic cosine law. The *discrete Gaussian curvature* is defined as the angle deficit,

$$K_i = \begin{cases} 2\pi - \sum \theta_i^{jk} & v_i \notin \partial M \\ \pi - \sum \theta_i^{jk} & v_i \in \partial M \end{cases}.$$

Circle packing metric. We associate each vertex v_i with a disk with radius γ_i. On edge $e_{ij} = [v_i, v_j]$, the two circles intersect at angle ϕ_{ij}. Then the edge length $l_{ij}$of e_{ij} is determined by the hyperbolic cosine law:

$$\cosh l_{ij} = \cosh \gamma_i \cosh \gamma_j + \sinh \gamma_i \sinh \gamma_j \cos \phi_{ij}. \quad (1)$$

A *circle packing metric* is denoted as (M, Γ, Φ), where $\Gamma : v_i \to \gamma_i$ represents the radius, $\Phi : e_{ij} \to \phi_{ij}$ represents the intersection angle. See Fig. 3 (b).

Let $u_i = \log \tanh \frac{\gamma_i}{2}$, *the discrete Ricci flow* is defined as

$$\frac{du_i(t)}{dt} = -K_i, \quad (2)$$

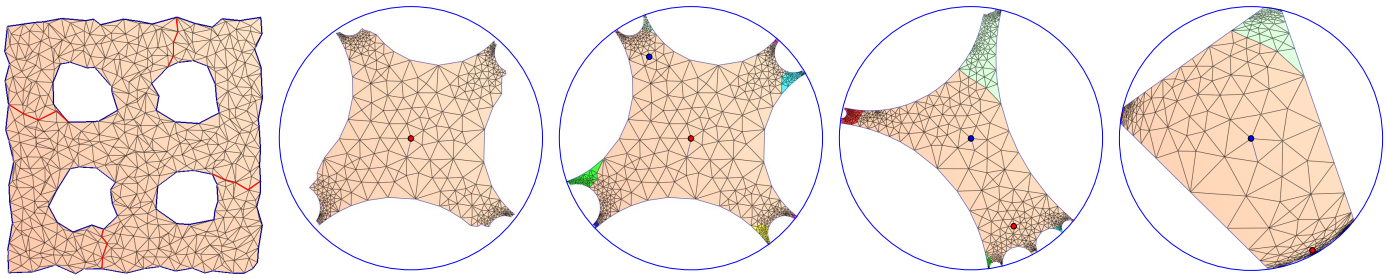
where K_i is the discrete Gaussian curvature at v_i.

The convergence of the discrete Ricci flow to the hyperbolic metric is proven by Chow and Luo [2]. The *Ricci energy* for circle packing metric (M, Γ, Φ) is defined as $E(\mathbf{u}) = \int_{\mathbf{u}_0}^{\mathbf{u}} \sum_{i=1}^{n} K_i du_i$, where $\mathbf{u}_0 = (0, 0, \cdots, 0)$, $\mathbf{u} = (u_1, u_2, \cdots, u_n)$. The discrete hyperbolic Ricci energy is convex. It has a unique global minimum, that induces the uniformization hyperbolic metric. Discrete Ricci flow in Eqn. 2 is the negative gradient flow of the Ricci energy. The gradient decent method (Eqn. 2) relies only on local information. At every iteration, a node needs to be only aware of the circle packing metric available from its neighbors. Therefore it admits a distributed algorithm. The details can be found in [22].

C. Embedding in Poinaré Disk

Once the hyperbolic metric is computed, we can embed the triangulation isometrically onto the Poincaré disk [11], [12].

1) Suppose a hyperbolic triangle has edge lengths $\{l_i, l_j, l_k\}$, then we compute the angle θ_k using the hyperbolic cosine law of Eqn. 1.



(a) homotopy group (b) fundamental domain (c) 1-ringed UCS (d) refocusing of (c) (e) Klein model picture of (d)

Fig. 5. Computing the hyperbolic embedding of a 5-connected domain with 525 nodes. (a) Compute a set of canonical homotopy group basis $\{a_1, a_2, a_3, a_4\}$; (b) Compute the hyperbolic uniformization metric using hyperbolic Ricci flow. The fundamental domain is isometrically embedded onto $\mathbb{H}^2$ under the hyperbolic metric; (c) Compute the Fuchsian group generators. Any finite portion of the universal covering space (UCS) can be constructed using these generators; (d) Refocus the UCS by Möbius transformation with specified origin point; (e) Embedding in Klein projective model, *convexity* is apparent to the eye. [27].

2) Set the coordinates of v_k to 0, those of v_i to $\cosh \frac{l_j}{2}$, those of v_j to $e^{i\theta_k} \cosh \frac{l_i}{2}$.
3) Glue the planar images of adjacent triangles by Möbius transformations. Suppose $f_1 = [v_i, v_j, v_k]$ and $f_2 = [v_j, v_i, v_l]$ are two adjacent triangles, the planar complex coordinates of v_i, v_j, v_k are z_i, z_j, z_k. We construct a Möbius transformation $\phi_1 : f_1 \to D$, such that $\phi_1(z_i)$ is the origin, $\phi_1(z_j)$ is on the real axis.

$$\tau_1 : z \to \frac{z - z_i}{1 - \bar{z}_i z},$$

then τ_1 maps z_i to 0, maps edge $[v_i, v_j]$ to a straight line. Then we construct another Möbius transformation

$$\tau_2 : z \to e^{i\theta} z,$$

where $\theta = arg\tau_1(z_j)$. Let $\phi_1 = \tau_2 \circ \tau_1 : f_1 \to D$, then ϕ_1 maps v_i to the origin, ϕ_1 maps v_j to be on the real axis. Similarly, we construct $\phi_2 : f_2 \to D$, which maps v_i to the origin, and v_j to be on the real axis. Then the Möbius transformation $\phi_2^{-1} \circ \phi_1$ glues the planar image of f_2 to the planar image of f_1 along edge $[v_i, v_j]$.

Observe that these are all local operations, requiring communications only between neighboring triangles. Thus, starting with an arbitrary node (say v_k) at 0, we can lay down the triangles in a distributed manner at the cost of a single flood.

One can further improve the computational accuracy for hyperbolic embedding with the methods in [11]. Figure 5 shows the hyperbolic embedding for a 5-connected domain, where the portion of UCS are constructed by gluing the fundamental domain to each cut boundary using suitable Möbius transformations.

D. Computing Fuchsian Group Generators

Let $\{c_1, c_2, \cdots, c_n\}$ be the cuts where S is sliced along c_k's to get $\tilde{S}$, each c_k is replicated to two boundary segments c_k^+ and c_k^- in a fundamental domain X in $\tilde{S}$. The embedding of all points on these two curves are supplied by the method in the previous subsection.

Corresponding to these two curves there are two Möbius transforms g_k^+ and g_k^-. These are the Fuchsian group generators associated with the cut c_k. Since this is an orientation preserving rigid transformation, the images in c_k^+ and c_k^- of any two points on c_k determines these two quantitie uniquely. Therefore, after the embedding is available, any pair of neighboring nodes on c_k can determine these generators completely locally. These can then be broadcast to the network.

E. Communication Cost

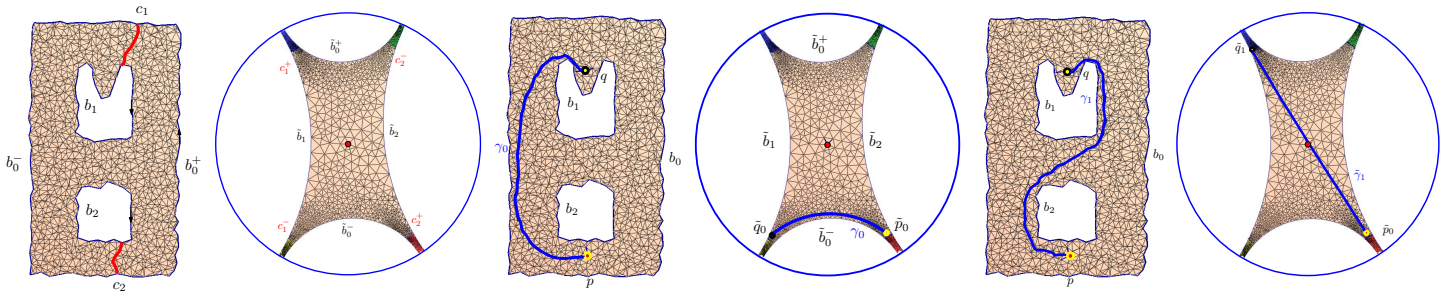
Last we summarize the communication cost involved in the construction of the hyperbolic embedding of the universal covering space. We measure the communication cost by the number of messages transmitted. The extraction of triangulation from the connectivity graph is a completely local algorithm with total messages in $O(n)$, where n is the network size. The step to slice the network holes open uses one round of flooding from each hole. The hyperbolic Ricci flow is an iterative algorithm. The number of iterations is evaluated in the simulation section. In the curvature flow, the vertex curvature satisfies the equation $K(t) = C_1 e^{-C_2 t}$, where C_1 and C_2 are two constants. The time complexity is given by $-C\frac{\log \epsilon}{\delta}$, where ϵ is the error tolerance, δ is the step length, C is a constant [2]. The embedding is obtained again by one round of flooding from an arbitrary root triangle. The nodes on a cut determine the generator locally, the total set of generators, whose number is proportional to the number of holes in the network, is disseminated to the entire network. Thus the total message cost, except the Ricci flow part, is $O(n)$ for a network with constant number of holes.

IV. Applications in Routing and Surveillance

In this section we discuss applications of the covering space embedding in routing and surveillance. But first we summarize the information obtained from the embedding method above in terminology suitable for application descriptions.

The embedding provides us with infinitely many copies of the network N in the hyperbolic space. Consider any one such copy X which we call a patch or a fundamental domain. Let us name with $X_1, X_2 \ldots$ other patches that are neighbors of X in the universal covering space $\tilde{N}$. The boundary between X and a neighboring patch X_i is the image of a *cut* that we used to obtain a simply connected domain. In fact the boundary of X contains two different images of each cut, separating X from two different neighboring patches.

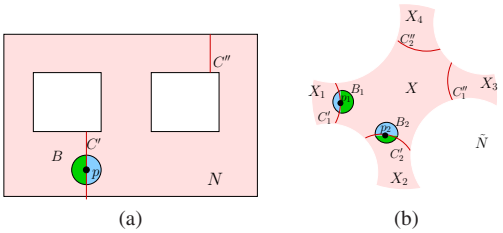
For example, consider the domain of Fig. 7 (a), with two holes. The two cuts C' and C'' are used to make the domain



(a) π-group (b) 1-ringed UCS (c) shortest path 1 (d) geodesic of (c) (e) shortest path 2 (f) geodesic of (e)

Fig. 6. Computing the shortest paths using the hyperbolic embedding of a 3-connected domain with 1286 nodes as shown in (a) with the homotopy group (π-group). The 1-ringed UCS in $\mathbb{H}^2$ is shown in (b). The source and target vertices p and q are given. The base path γ_0 from p to q is shown in (c). The geodesic homotopic to γ_0 is shown in (d). Another path γ_1 from p to q is in (e). The homotopy type of γ_1 ($[\gamma_1\gamma_0^{-1}]$) is a_1^{-1}.

simply connected. In the interior of X every other point of N occurs exactly once. The cut C' has two images C'_1 and C'_2 on the boundary of X, separating X from X_1 and X_2 respectively (Fig. 7 (b)). A point p on C' will have two images p_1 and p_2 on the two respective boundaries. A neighborhood B of p appears as two disjoint pieces in X, neighboring ν'_1 and p'_2 respectively.



(a) (b)

Fig. 7. A point p on a cut C' has two images p_1 and p_2 on the boundary of a fundamental domain. The ball B around p gets split by the cut and appears in two pieces in X. This does not imply any discontinuity of the map h^{-1} from N to $\tilde{N}$. The respective other half of B always appears in the neighboring patch thus ensuring local homeomorphism of the map.

Relation to Fuchsian generators. Suppose that $\{g_1, g_2 \dots\}$ are the generators of the Fuchsian group. This means that the transformation g_i maps X to X_i isometrically. X can be mapped to any other patch in the covering space by application of a suitable sequence of these generators. Further, for each g_i there is an inverse generator $g_j = g_i^{-1}$, so that g_j maps X_i to X. In the example of Fig. 7, $g_1 = g_2^{-1}$ and $g_3 = g_4^{-1}$. Therefore, $g_2(p_1) = p_2$ and $g_1(p_2) = p_1$.

Information from embedding phase. The embedding method provides us with sufficient information to create the entire covering space of the network. In particular, we have:

1) At each node, its coordinate in a particular fundamental domain X.
2) For a node p on a cut path, two different sets of coordinates p_1 and p_2 as described above.
3) The set of generators $G = \{g_1, g_2 \dots\}$ of the Fuchsian group.

With this background, we are ready to describe how to use this embedding information in sensor network applications.

A. Routing

First of all, note that given a coordinate for each node in a fundamental domain X, greedy routing can be used to route to any other point in X. Since X is convex in the hyperbolic plane, greedy routing using the hyperbolic metric always succeeds. However, it is not always desirable to route strictly within X as such a path may be unnecessarily long. Consider for example a small neighborhood B in Fig. 7 (a). It is possible that images in X of two points in this region are quite far (Fig. 7 (b)). It is true however, that the image in X of one such point must be within a small distance of the other point either in X_1 or in X_2. Therefore, we can find a short path by routing to some other image of the destination in the covering space.

Routing in covering space. We discuss the routing method that takes as input nodes (a, b) and coordinates $(a|_X, b|_Y)$, implying a query to route from the image of node a in X to the image of node b in patch Y. Recall that $b|_Y$ is obtained by applying to $b|_X$ the sequence of g_i's that map X to Y.

The routing method alternates between two phases:

1) **Greedy routing.** From the 1-hop neighborhood of a in the triangulation of the network, find q such that hyperbolic distance $d(q|_X, b|_Y)$ is minimized. If a itself is the minimizing node, we say this step has failed, and use phase 2, otherwise the routing proceeds from $q|_X$.
2) **Crossing the boundary.** Phase 1 can fail only at the boundary between X and X_i. That is, at a point like p in Fig. 7. Without loss of generality, let us say, it fails at $p_1|_X$. The neighbors of p in the triangulation are divided into two sets that can be labeled as neighbors of $p_1|_X$ and neighbors of $p_2|_X$ respectively. We find q such that $q|_X$ is a neighbor of $p_2|_X$, and from there execute the routing query $[(q, b), (q|_{X_1}, b|_Y)]$.

A few comments are in order. First of all, we have described the method using the triangulation graph for simplicity of description, the method can be operated on the UDG or quasi-UDG as well. In the results we present in the simulation section, the routing was done on the network graph, and not on the triangulation. Secondly, We have claimed that phase 1 fails only at boundaries of patches, never at network boundaries. This is true because in the embedding we obtain, not only is a patch a convex region, the entire covering space is convex in the hyperbolic metric. Thus, the geodesic realizing the shortest distance to destination must lie inside the covering space.

Choice of routing trajectory. We now have a method to route to any image of the destination. It is however not clear which such image to select. In general this choice will depend on the

needs of the application, we provide a discussion here to aid such decisions.

Consider nodes (a, b), and an arbitrary path from $a|_X$ to $b|_Y$ for an arbitrary patch Y. It is the property of the covering space that a unique choice of Y determines a unique homotopy type for the projected path in the original network. The homotopy type can be represented unambiguously by the sequence of g_is taking X to Y. An appearance of g_i in the sequence implies that at some point the path crosses from Z to a neighboring patch Z_i, crossing the corresponding cut. An appearance of $g_j = g_i^{-1}$ implies crossing the cut in the opposite direction, that is, in the covering space crossing over to Z_j. If g_i or g_i^{-1} does not appear in the sequence, the cut is never crossed.

Such sequences can be arbitrarily long in general, but we are often interested in sequences of some finite length k'. Figure 2 shows the cases for $k' = 2$. In a network, we are often interested in the sequences where each generator appears zero or one time. A generator appearing 2 or more times simply means the path is going around a hole in cycles, this is not useful in most routing scenarios. This leaves us with a finite number possibilities. In a network with k holes, this number is at least $O(2^k)$, since each g_i may appear zero or one times. Note that we use only $O(k)$ storage to select from a set that is exponential in k. We can additionally restrict the sequence to k' length. Thus, we select the destination image $b|_Y$ by applying such sequences to $b|_X$ and selecting appropriately. In the simulations in the next section, we select $k' = 1$ and select $b|_Y$ that minimizes the hyperbolic distance $d(a|_X, b|_Y)$.

B. Cycle contractibility

Given a cycle in the network, we want to test whether it surrounds one or multiple holes. The test is based on the following fact. If γ is a closed curve in N and $\tilde{\gamma}$ is a lift of γ to the covering space $\tilde{N}$, then γ is contractible if and only if $\tilde{\gamma}$ is a closed cycle. Therefore, given a cycle in the network, we move hop-by-hop along the cycle, and have a pointer that moves correspondingly in the covering space. The cycle is contractible if and only if the pointer returns to the starting point when the cycle ends.

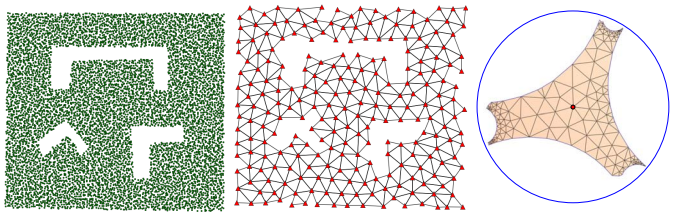
We can also generate cycles that surround one or more holes. As we know, corresponding to each hole, there is a cut c connecting it to the outer boundary. Correspondingly, there are Fuchsian generators g, g^{-1}. To create a cycle starting at node s and surrounding a set of k holes, we just need to apply the corresponding generators to s getting $s' = g_1 \dots g_k(s)$ and find a path from s to s'.

This knowledge of the generators allows us to create cycles of arbitrary homotopy types, and is a simple matter to generalize to cases where all cuts do not connect to the outer boundary. We omit the details here.

V. Simulations

We carried out simulations on the graph of the network in Fig. 8 (a), and on networks based on similar geometry but different numbers of nodes. In particular, we tested the properties of the routing method described in the previous section. The following are the major conclusions in that respect.

- The routing method successfully delivered in 100% of cases, there were no failures.
- The routing stretch (ratio of path length to the length of shortest path) was small on average, only 1.15.
- The traffic load was evenly balanced among nodes compared to other greedy routing methods.



(a) (b) (c)

Fig. 8. (a) Network with 3 holes and of approximately 8700 nodes, distributed in a perturbed grid, average degree of about 20 in quasi-UDG. (b) Triangulation of network obtained by landmark triangulation method, without using node geographical locations. (c) embedding of the triangulation.

Load balancing and stretch. We ran routing queries on $10,000$ randomly selected source-destination pairs, and compared load balancing properties with Kleinberg's method [14] of embedding a spanning tree of the network in hyperbolic space and shortest path routing. In Table I, *load* represents the total number of messages a node has to handle. The covering space embedding has better load balancing properties is further borne out by the plots in Fig. 9.

TABLE I
LOAD COMPARISON

Method	Average Load	Max Load
Covering space embedding (ours)	23.37	368
Spanning tree embedding [14]	32.92	1918
Shortest path routing	19.44	538

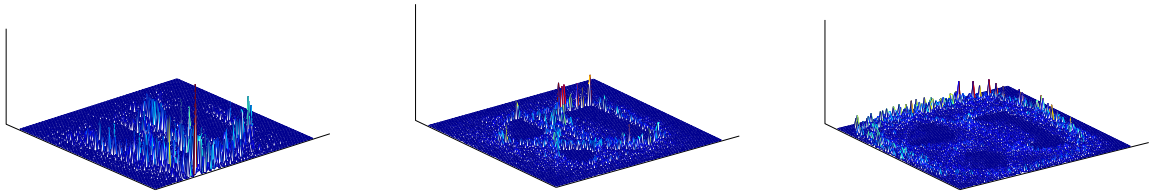
Other greedy routing methods such as [22] tend to hug the boundary and thus produce uneven load similar to shortest path routes. One way to interpret the high load in spanning tree embedding method is that the spanning tree causes many *cuts* in the network such that neighbors across the cut may be quite far in the embedding. And unlike the covering space embedding, it does not restore the continuity of embedding across the cuts.

We measured the stretch on the paths, and found that the stretch was remarkably small, only about 1.15, whereas the stretch in the spanning tree embedding method is much higher — about 1.78, while the method in [22] has a stretch of 1.59.

Convergence time. We carried out experiments to test the number of iterations of the distributed hyperbolic Ricci flow to convergence.

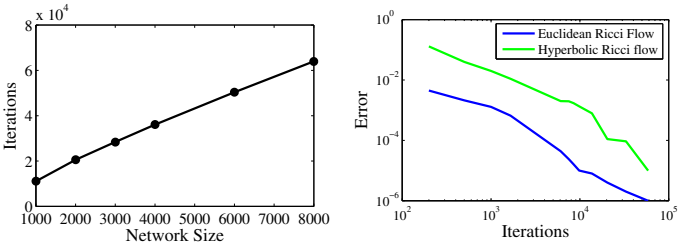
The results in Fig. 10 show that while the hyperbolic Ricci flow scales linearly with network size, it is somewhat slower than the Euclidean Ricci flow.

We also conducted experiments on using Newton's numerical method to compute the solution. In this centralized method, the computation is very efficient, and error reduces to 10^{-8} in as few as 12 iterations. As mentioned earlier, it is possible to obtain an embedding without node locations, making use



(a) (b) (c)

Fig. 9. Comparison of load distributions. (a) Load distribution in the spanning tree embedding method. Load is generally high, and is particularly high at and around the center of the tree. The tree was generated as the shortest path tree stating at a random node. This is a simple distributed way to obtain spanning trees in sensor networks; (b) Load distribution in shortest path routing. Load is seen to be higher along the boundaries and at the center; (c) Load distribution in covering space embedding is better than other methods.



(a) (b)

Fig. 10. Convergence times. (a) The number of iterations to reach a curvature error of 10^{-6} for networks of different sizes, but of the same topology as Fig. 8. The growth is clearly close to linear. (b) The convergence rate of curvature.

of triangulation on landmarks. We carried out some such operations and corresponding images are shown in Fig. 8.

VI. Conclusion

In this paper we proposed to use the embedding of the universal covering space of the sensor network in a hyperbolic space for resilient routing. In particular, one can find routes of a particular homotopy type with simple greedy methods. We also demonstrated that the greedy routing in hyperbolic space has 100% delivery and improves load balancing as the routes naturally avoid the hole boundaries. For our future work we would like to investigate further the application of the hyperbolic embedding in load balancing with provable results, as well as applications in mobile networks [20].

Acknowledgements. This work is partially supported by NSFCCF448399, ONRN00140910228, and NSFC60628202. J. Gao and R. Sarkar are partially supported by NSF CAREER Award CNS-0643687.

References

[1] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.

[2] B. Chow and F.Luo. Combinatorial Ricci flows on surfaces. *Journal of Differential Geometry*, 63(1):97–129, 2003.

[3] Q. Fang, J. Gao, and L. Guibas. Locating and bypassing routing holes in sensor networks. In *Mobile Networks and Applications*, volume 11, pages 187–200, 2006.

[4] R. Flury, S. Pemmaraju, and R. Wattenhofer. Greedy routing with bounded stretch. In *INFOCOM*, 2009.

[5] S. Funke and N. Milosavljević. Guaranteed-delivery geographic routing under uncertain node locations. In *Proceedings of the 26th Conference of the IEEE Communications Society (INFOCOM'07)*, pages 1244–1252, May 2007.

[6] S. Funke and N. Milosavljević. Network sketching or: "how much geometry hides in connectivity? - part II". In *SODA '07: Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 958–967, 2007.

[7] D. Ganesan, R. Govindan, S. Shenker, and D. Estrin. Highly-resilient, energy-efficient multipath routing in wireless sensor networks. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(4):11–25, 2001.

[8] J. Gao, L. J. Guibas, J. Hershberger, L. Zhang, and A. Zhu. Geometric spanners for routing in mobile networks. *IEEE Journal on Selected Areas in Communications Special issue on Wireless Ad Hoc Networks*, 23(1):174–185, 2005.

[9] R. S. Hamilton. Three manifolds with positive Ricci curvature. *Journal of Differential Geometry.*, 17:255–306, 1982.

[10] R. S. Hamilton. The Ricci flow on surfaces. *Mathematics and general relativity*, 71:237–262, 1988.

[11] M. Jin, J. Kim, F. Luo, and X. Gu. Discrete surface Ricci flow. *IEEE TVCG*, 14(5):1030–1043, 2008.

[12] M. Jin, F. Luo, and X. Gu. Computing surface hyperbolic structure and real projective structure. In *SPM'06: Proceedings of the 2006 ACM Symposium on Solid and Physical Modeling*, pages 105–116, 2006.

[13] B. Karp and H. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 243–254, 2000.

[14] R. Kleinberg. Geographic routing using hyperbolic space. In *Proceedings of the 26th Conference of the IEEE Communications Society (INFOCOM'07)*, pages 1902–1909, 2007.

[15] R. Musaloiu-E. and A. Terzis. Minimising the effect of wifi interference in 802.15.4 wireless sensor networks. *International Journal of Sensor Networks*, 3(1):43–54, 2008.

[16] G. Perelman. The entropy formula for the ricci flow and its geometric applications. Technical Report arXiv.org, November 11 2002.

[17] G. Perelman. Finite extinction time for the solutions to the ricci flow on certain three-manifolds. Technical Report arXiv.org, July 17 2003.

[18] G. Perelman. Ricci flow with surgery on three-manifolds. Technical Report arXiv.org, March 10 2003.

[19] A. Rao, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 96–108, 2003.

[20] N. S. V. Rao, Q. Wu, S. S. Iyengar, and A. Manickam. Connectivity-through-time protocols for dynamic wireless networks to support mobile robot teams. In *International Conference on Robotics and Automation 2003*, volume 2, pages 1653–1658, 2003.

[21] J. R.Munkres. *Elements of Algebraic Topology*. Addison-Wesley Co., 1984.

[22] R. Sarkar, X. Yin, J. Gao, F. Luo, and X. D. Gu. Greedy routing with guaranteed delivery using ricci flows. In *Proc. of the 8th International Symposium on Information Processing in Sensor Networks (IPSN'09)*, April 2009.

[23] R. Schoen and S.-T. Yau. *Lectures on Differential Geometry*. International Press of Boston, 1994.

[24] Y. Wang, J. Gao, and J. S. B. Mitchell. Boundary recognition in sensor networks by topological methods. In *Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 122–133, September 2006.

[25] W. Xu, K. Ma, W. Trappe, and Y. Zhang. Jamming sensor networks: attack and defense strategies. *Network, IEEE*, 20(3):41–47, May-June 2006.

[26] W. Zeng, M. Jin, F. Luo, and X. Gu. Canonical homotopy class representative using hyperbolic structure. In *IEEE International Conference on Shape Modeling and Applications (SMI'09)*, pages 171–178, 2009.

[27] W. Zeng, D. Samaras, and X. D. Gu. Ricci flow for 3D shape analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.